The Consultative Committee for Space Data Systems

Draft Recommendation for
Space Data System Practices

SPACECRAFT ONBOARD INTERFACE SERVICES— TIME ACCESS SERVICE

DRAFT RECOMMENDED PRACTICE

CCSDS 872.0-R-1

RED BOOK
June 2007

AUTHORITY

Issue:	Red Book, Issue 1
Date:	June 2007
Location:	Not Applicable

(WHEN THIS RECOMMENDED PRACTICE IS FINALIZED, IT WILL CONTAIN THE FOLLOWING STATEMENT OF AUTHORITY:)

This document is published and maintained by:

CCSDS Secretariat
Office of Space Communication (Code M-3)
National Aeronautics and Space Administration
Washington, DC 20546, USA

STATEMENT OF INTENT

(WHEN THIS RECOMMENDED PRACTICE IS FINALIZED, IT WILL CONTAIN THE FOLLOWING STATEMENT OF INTENT:)

The Consultative Committee for Space Data Systems (CCSDS) is an organization officially established by the management of its members. The Committee meets periodically to address data systems problems that are common to all participants, and to formulate sound technical solutions to these problems. Inasmuch as participation in the CCSDS is completely voluntary, the results of Committee actions are termed **Recommendations** and are not in themselves considered binding on any Agency.

CCSDS Recommendations take two forms: **Recommended Standards** that are prescriptive and are the formal vehicles by which CCSDS Agencies create the standards that specify how elements of their space mission support infrastructure shall operate and interoperate with others; and **Recommended Practices** that are more descriptive in nature and are intended to provide general guidance about how to approach a particular problem associated with space mission support. This **Recommended Practice** is issued by, and represents the consensus of, the CCSDS members. Endorsement of this **Recommended Practice** is entirely voluntary and does not imply a commitment by any Agency or organization to implement its recommendations in a prescriptive sense.

No later than five years from its date of issuance, this **Recommended Practice** will be reviewed by the CCSDS to determine whether it should: (1) remain in effect without change; (2) be changed to reflect the impact of new technologies, new requirements, or new directions; or (3) be retired or canceled.

In those instances when a new version of a **Recommended Practice** is issued, existing CCSDS-related member Practices and implementations are not negated or deemed to be non-CCSDS compatible. It is the responsibility of each member to determine when such Practices or implementations are to be modified. Each member is, however, strongly encouraged to direct planning for its new Practices and implementations towards the later version of the Recommended Practice.

FOREWORD

(WHEN THIS RECOMMENDED PRACTICE IS FINALIZED, IT WILL CONTAIN THE FOLLOWING FOREWORD:)

Through the process of normal evolution, it is expected that expansion, deletion, or modification of this document may occur. This Recommended Practice is therefore subject to CCSDS document management and change control procedures, which are defined in the *Procedures Manual for the Consultative Committee for Space Data Systems*. Current versions of CCSDS documents are maintained at the CCSDS Web site:

http://www.ccsds.org/

Questions relating to the contents or status of this document should be addressed to the CCSDS Secretariat at the address indicated on page i.

PREFACE

This document is a draft CCSDS Recommended Practice. Its draft status indicates that the CCSDS believes the document to be technically mature and has released it for formal review by appropriate technical organizations. As such, its technical contents are not stable, and several iterations of it may occur in response to comments received during the review process.

Implementers are cautioned **not** to fabricate any final equipment in accordance with this document's technical content.

DOCUMENT CONTROL

Document	Title	Date	Status
CCSDS 872.0-R-1	Spacecraft Onboard Interface Services—Time Access Service, Draft Recommended Practice, Issue 1	June 2007	Current draft

CONTENTS

1 INTRODUCTION

1.1 PURPOSE AND SCOPE

This document defines the SOIS Time Access Service. The Time Access Service provides a local copy of the spacecraft onboard time which is correlated across all of the nodes implementing the service. This time can be used, for example, for coarse-grained scheduling of onboard operations and for time stamping of acquired data.

The Time Access Service is not intended to be used as a fine-grained, multi-purpose timing mechanism for applications; e.g., it should not be used for software task scheduling.

The benefit of using the Time Access Service is that onboard software applications have a consistent interface to the local time source regardless of how time correlation is actually performed.

1.2 DOCUMENT STRUCTURE

This document comprises three sections:

– section 1, this section, defines common terms used within this document and lists reference documents;

– section 2 (informative) describes the Time Access Service concept;

– section 3 (normative) defines the Time Access Service, in terms of the service provided, services expected from underlying layers, and the service interface.

In addition, three informative annexes are provided:

– annex A describes how local time registers are implemented and how the master onboard time value is usually distributed in current spacecraft;

– annex B describes aspects of the POSIX API time functions which might be useful to implementers of the Time Access Service;

– annex C contains a list of informative references.

1.3 DEFINITIONS

1.3.1 DEFINITIONS FROM THE OSI REFERENCE MODEL

The command and data acquisition service is defined using the style established by the Open Systems Interconnection (OSI) Basic Reference Model (reference [1]). This model provides a common framework for the development of standards in the field of systems interconnection.

The following terms used in this Recommended Practice are adapted from definitions given in reference [1]:

layer: a subdivision of the architecture, constituted by subsystems of the same rank.

service: a capability of a layer, and the layers beneath it (a service provider), which is provided to the service users at the boundary between the service providers and the service users.

1.3.2 TERMS DEFINED IN THIS RECOMMENDED PRACTICE

For the purposes of this Recommended Practice, the following definitions also apply.

Accuracy: The closeness of the agreement between the times reported by a local time source and the master onboard time source of the spacecraft.

Application: Any component of the onboard software that makes use of this service. This includes flight software applications and higher-layer services.

Error bound: An indication of the notional error range of a particular time value. Larger values for the error bound indicate less accuracy in the returned time.

Precision: The smallest significant time increment that can be reported by a time source. For example, a time source that reports time as *minutes:seconds:milliseconds* has a precision of 1ms. Note that the precision of a time source may not be the same as the resolution of that time source.

Resolution: The smallest significant time increment that can be measured by a time source. For example, a time source driven by a 1MHz oscillator has a resolution of 1µs.

Time correlation: The process of maintaining coherence between the master time source and each local time source, such that all time sources provide the same time value within some bounded uncertainty. This maximizes the accuracy of each time source with respect to the master onboard time source.

1.4 DOCUMENT NOMENCLATURE

The following conventions apply throughout this Recommended Practice:

a) The words ‘shall’ and ‘must’ imply a binding and verifiable specification;

b) The word ‘should’ implies an optional, but desirable, specification;

c) The word ‘may’ implies an optional specification;

d) The words ‘is’, ‘are’, and ‘will’ imply statements of fact.

1.5 REFERENCES

The following documents contain provisions which, through reference in this text, constitute provisions of this Recommended Practice. At the time of publication, the editions indicated were valid. All documents are subject to revision, and users of this Recommended Practice are encouraged to investigate the possibility of applying the most recent editions of the documents indicated below. The CCSDS Secretariat maintains a register of currently valid CCSDS Documents.

[1] *Information Technology—Open Systems Interconnection—Basic Reference Model: The Basic Model*. International Standard, ISO/IEC 7498-1. 2nd ed. Geneva: ISO, 1994.

NOTE – Informative references are contained in annex C.

2 SERVICE CONCEPT

2.1 OVERVIEW

The SOIS Time Access Service (TAS) is defined within the context of the overall SOIS architecture (reference [C5]) as one of the services of the Application Support Layer, as illustrated in the following figure.

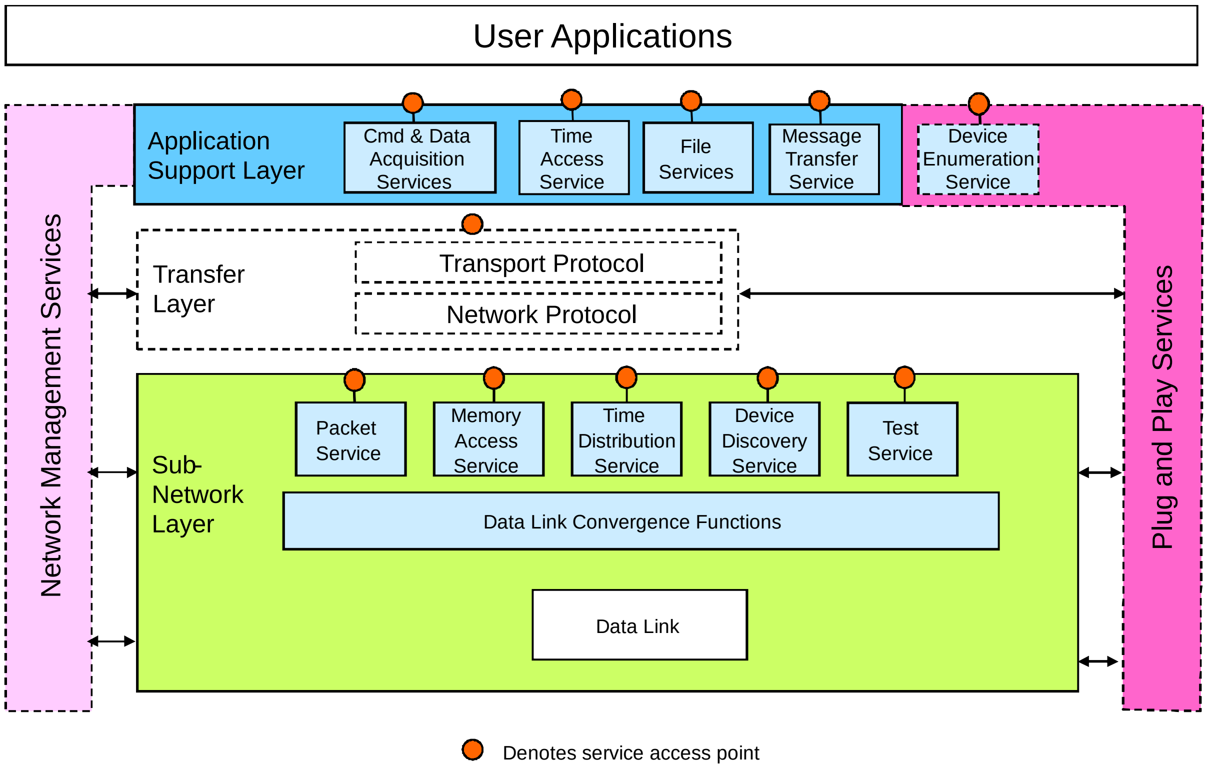

Figure 2-1: SOIS Time Access Service Context

NOTE – The SOIS Time Access Service is one of the services of the Application Support Layer of the SOIS Architecture.

The SOIS Time Access Service provides applications with a consistent interface to a local time source that is correlated to some centrally maintained master onboard time source. The time values provided by this service might typically be used by the application to schedule some operation, such as the acquisition of an image or to time stamp locally generated telemetry data.

The need to provide a local, correlated time source in onboard processor nodes is common to all spacecraft that have more than one processing node connected to an onboard bus or LAN. A typical architectural scenario is shown in figure 2-2. Note that the SOIS Time Access Service is concerned only with providing the interface to the local time source. It is not concerned with the mechanism used to correlate the time between the time sources, which is addressed by the SOIS Time Distribution Service (reference [C6]).

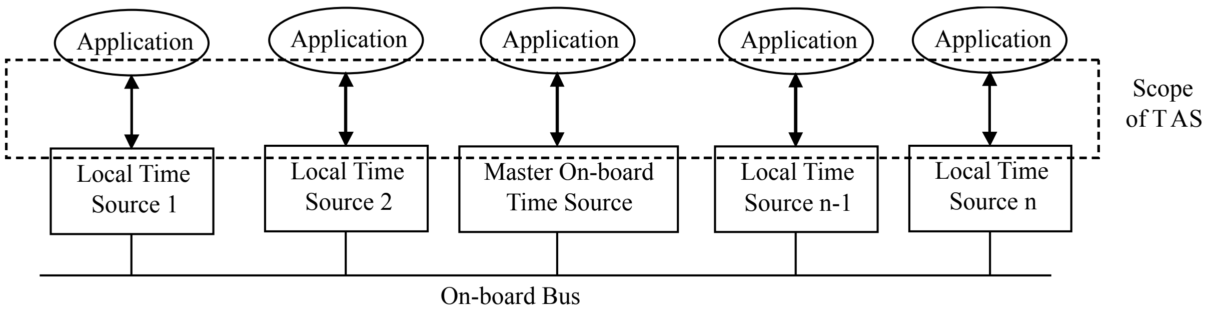

Figure 2-2: Typical Onboard Time System Architecture

NOTE – A typical onboard time system architecture consists of local and master onboard time sources implemented in hardware. The SOIS Time Access Service provides access to a local time source only.

In this architecture the local time sources are typically free-running hardware counters accumulating seconds and sub-seconds of elapsed time. Each of these counters is driven by its own oscillator, and the absolute frequency and frequency stability of each oscillator are different in each node. The master onboard time source, the reference for onboard time for all onboard mission operations, is usually a similar free-running counter driven by an oscillator with precise absolute frequency and high stability.[1] The value provided by this time source is usually called the Mission Elapsed Time (MET).

The actual method of implementing the master onboard time source and the local time sources and the mechanisms used to correlate them is outside the scope of this document. However, annex A describes a scheme that is typically used today.

This document defines a standard interface between applications hosted on each node and the local time source for that node. The scope of this document is indicated by the dashed box in figure 2-2. The basic capability provided by the SOIS Time Access Service is the ability to read the time on demand, i.e., a '*wall clock*' capability. Two optional extensions are also defined which reflect common requirements for onboard software systems. The first of these is an '*alarm clock*' capability, which enables the application to request notification at a particular time. The second is a '*metronome*' capability, which enables the application to request periodic notifications with a specified interval and starting at a particular time.

The benefit to the user is that all applications have a uniform interface to the local time source, regardless of their location on the spacecraft, and do not have to access local hardware directly. This simplifies the development of the applications and means that they can be relocated if necessary and can be re-used in other missions.

[1] In certain orbits it is possible to use a time source that is correlated to an external reference time source, such as a GPS receiver, as the master onboard time source.

2.2 USE AND OPERATION OF THE TIME ACCESS SERVICE

Applications should use the Time Access Service to obtain the time from the local time source rather than, for example, reading directly from the local elapsed time counter hardware registers. From the application software perspective this will result in applications that are more portable, easier to develop, and independent of the hardware implementations of the onboard time sources.

This time can be used, for example, for coarse-grained scheduling of onboard operations and for time stamping of acquired data.

The Time Access Service is not intended to be used as a fine-grained, multi-purpose timing mechanism for applications; e.g., it should not be used for software task scheduling.

The Time Access Service is operated using service requests and service indications passed between the service user and the service provider.

3 SERVICE DEFINITION

3.1 PROVIDED SERVICE

3.1.1 WALL CLOCK CAPABILITY (MANDATORY)

The mandatory wall clock capability shall allow the user to obtain the local onboard time on demand.

To obtain the time, the user shall issue a TAS_TIME.request primitive. The service shall respond with a TAS_TIME.indication primitive that contains the current local time and an indication of the accuracy and validity of this time value.

3.1.2 ALARM CLOCK CAPABILITY (OPTIONAL)

The optional alarm clock capability shall allow the user to register for an alarm call at a specified time.

To register for an alarm call, the user shall issue a TAS_ALARM.request primitive with the time (absolute or relative) at which the alarm call is to be issued. The service shall respond with a TAS_TIME.indication primitive issued at the specified time or immediately indicating failure of the registration. The user may cancel a pending alarm call by issuing a TAS_CANCEL_ALARM.request.

3.1.3 METRONOME CAPABILITY (OPTIONAL)

The optional metronome capability shall allow the user to register for a periodic indication of the time at a specified frequency and starting at a specific time.

To register for a periodic indication of the time, the user shall issue a TAS_METRONOME.request primitive with the time at which the first periodic indication is to be issued and frequency at which subsequent indications are to be. The service shall respond with periodic TAS_TIME.indication primitives with the first indication being issued at the specified start time and subsequent indications being issued at the specified frequency. The service shall respond immediately with a TAS_TIME.indication primitive to indicate failure of the registration. To stop the metronome, the user shall issue a TAS_CANCEL_METRONOME.request.

3.2 EXPECTED SERVICES FROM UNDERLYING LAYERS

The expected service from the underlying layers is a locally maintained clock that indicates seconds and sub-seconds of monotonically increasing elapsed time.

Note however that this does not preclude the locally maintained clock's being 'wound back', e.g., because if an inaccuracy causing it to run fast. This may be expressed as an elapsed time from some specific epoch, such as the start of the mission, or may be a real-time clock. The maximum adjustment permissible per request will be defined in the MIB, so as to minimise the affect upon the service users.

3.3 SERVICE INTERFACE

3.3.1 GENERAL

The Time Access Service interface comprises the following primitives:

– TAS_TIME.request;

– TAS_ALARM.request;

– TAS_CANCEL_ALARM.request;

– TAS_METRONOME.request;

– TAS_CANCEL_METRONOME.request;

– TAS_TIME.indication.

These primitives and their associated parameters are described in the following sections.

3.3.2 TAS_TIME.REQUEST

The TAS_TIME.request shall be issued by the Time Access Service user in order to request the current time. Receipt of this primitive shall cause the Time Access Service to determine the current time.

There are no parameters associated with this primitive.

3.3.3 TAS_ALARM.REQUEST

The TAS_ALARM.request shall be issued by the Time Access Service user in order to request a one-shot alarm at a specific time in the future.

The parameter associated with this primitive is:

– Alarm_at_time.

Alarm_at_time is the time at which the user wishes to receive a time indication.[2]

[2] Mechanisms to associate particular TAS_TIME.indications with their corresponding TAS_ALARM.requests are implementation-specific. Similar mechanisms need to be provided by implementations to allow the TAS_CANCEL_ALARM.request to be associated with a particular TAS_ALARM.request, and a TAS_CANCEL_METRONOME.request to be associated with a particular TAS_METRONOME.request.

3.3.4 TAS_CANCEL_ALARM.REQUEST

The TAS_CANCEL_ALARM.request shall be issued by the Time Access Service user in order to request that a previously requested alarm call be cancelled. This shall cause the cancellation of the scheduled alarm call and removal of all state associated with it.

There are no parameters associated with this primitive.[2]

3.3.5 TAS_METRONOME.REQUEST

The TAS_METRONOME.request shall be issued by the Time Access Service user in order to request a periodic time indication at a specific time in the future and at specified intervals thereafter.

The parameters associated with this primitive are:[2]

– First_alarm_time;

– Inter_alarm_interval.

First_alarm_time is the time at which the user wishes to receive the first of a series of periodic alarms.

Inter_alarm_interval is the inter-alarm time interval between successive alarms.

3.3.6 TAS_CANCEL_METRONOME.REQUEST

The TAS_CANCEL_METRONOME.request shall be issued by the Time Access Service user in order to request that a previously requested metronome be cancelled. This shall cause the cancellation of the scheduled or active metronome and removal of all state associated with it.

There are no parameters associated with this primitive.[2]

3.3.7 TAS_TIME.INDICATION

The TAS_TIME.indication shall be issued by the Time Access Service in response to a TAS_TIME.request, TAS_ALARM.request, or a TAS_METRONOME.request in order to deliver the current time, and shall indicate whether the request was executed successfully or not.

The parameters associated with this primitive are:

– Current_time;

– Result;

– Current_time_error_specification.

Current_time is the current time at this node, as determined by the Time Access Service.

Result indicates whether the request was executed correctly or not. A *No_Error* result implies that the time was successfully obtained and the associated Current_time parameter is valid. Other results indicate failure conditions, e.g., *Timeout* (the specified request could not be serviced within the managed timeout period) or *Error* (the time access service is not functioning correctly, in which case the associated Current_time parameter is not valid).

Current_time_error_specification indicates the error of the Current_time parameter returned in the time indication.

4 MANAGEMENT INFORMATION BASE

The **Frequency**[3] is a managed parameter that indicates the rate, in Hertz, at which the Time Access Service is updated. Note this is not the frequency of a local time source, e.g., hardware clock. It is the frequency at which the TAS samples it. A service management entity should be able to access this parameter, but it is usually not possible to update it when the local time source is implemented in hardware.

The **Drift** is a managed parameter that indicates the stability of the local oscillator used in the local clock. A service management entity should be able to access this parameter and should be able to update it to compensate for clock aging effects.

The **Maximum Adjustment** is a managed parameter that indicates the maximum permissible adjustment, forward or backward, that can be made to the local time source in a single request.

[3] This parameter is defined in terms of frequency rather than interval. While this is inconsistent with the rest of the document, the terminology is retained since it is more usual to refer to the frequency of a local time source.

5 SERVICE CONFORMANCE STATEMENT PROFORMA

It is mandatory that, for any implementation claiming to provide this service, this proforma be completed giving details of the capabilities of the implementation.

Service Conformance Statement

SOIS Time Access Service

Implementation Information

Implementer Identification	
Implementation Identification	
Version	
Underlying Data link	
Protocol Specification Reference	
MIB Reference	

Mandatory Features

TAS_TIME.request	√
TAS_TIME.indication	√

Optional Features

TAS_ALARM.request	
TAS_CANCEL_ALARM.request	
TAS_METRONOME.request	
TAS_CANCEL_METRONOME.request	

Other Information

N/A	

ANNEX A

TYPICAL IMPLEMENTATION

(Informative)

This annex describes a hardware-based implementation scheme for local time sources that is commonly used in modern spacecraft. This description is provided for information only.

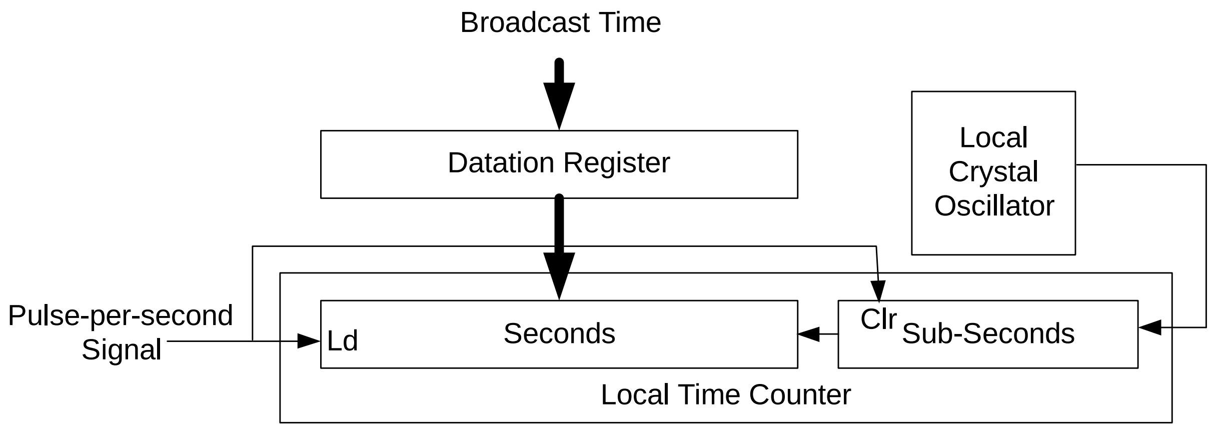

Figure A-1: Typical Local Time Source Implementation

A typical local time source implementation comprises a free-running counter clocked by a local crystal oscillator. Typically, the oscillator frequency and counter configuration are chosen such that the elapsed time accumulated by the counter is related to an exact number of seconds and sub-seconds. For example, a 32-bit counter might be used with 24-bits corresponding to whole seconds and eight-bits of sub-seconds. This time source would have a precision of just under 4ms and a roll-over period of about 194 days. This time source format is convenient for scheduling onboard operations and also matches the CCSDS Unsegmented Time Code (CUC) format often used for telemetry packet time stamping (see reference [C3]).

Left to its own devices this local time source would record the elapsed time since the unit was powered on. However, the local oscillator has particular frequency accuracy and drift characteristics that will differ from other local time source oscillators on the spacecraft. Furthermore, the flight units may be powered on at different times during the mission. Consequently, each node will report a different value of elapsed time according to the local parameters. Therefore there is usually a requirement to select a single onboard time source as the reference source, or master, for the mission elapsed time, and to implement a mechanism that correlates all of the local time sources so that they report the same time value within reasonable bounds determined by the mission requirements.

Typically this is done by periodically broadcasting the reference source time value via the spacecraft data handling bus and issuing a precisely timed reference pulse indicating exactly when this time is valid. A common implementation, for example, is to broadcast the whole seconds value of the reference mission elapsed time value on a MIL-STD-1553B bus and distribute a dedicated pulse-per-second (PPS) signal to all nodes indicating when this time becomes valid. This is analogous to the talking clock service; ‘at the first stroke it will be 10:58 precisely’.

The way this is handled in the local time source implementation is to add a special *datation* register that is loaded with the broadcast time value received on the spacecraft bus as shown in figure A-1**.** This value is then transferred into the whole seconds’ section of the local elapsed time counter on receipt of the PPS signal. At the same instant the sub-seconds portion of the local counter is cleared. Thus the local time source is accurately correlated to the reference mission elapsed time exactly once per second. Following this instance, the local time source will drift according to the characteristics of the local oscillator until, just before the next PPS signal, it can be expected to reach its maximum deviation with respect to this reference.

Depending on the sophistication of the local time source, it can detect missed correlation steps and can be configured such that the correlation operation never causes the indicated time to run backwards. These techniques are beyond the scope of this brief description.

ANNEX B

TYPICAL API – POSIX

(Informative)

NAME[4]

time.h - time types

SYNOPSIS

#include <time.h>

DESCRIPTION

Some of the functionality described on this reference page extends the ISO C standard. Applications shall define the appropriate feature test macro (see the System Interfaces volume of IEEE Std 1003.1-2001, Section 2.2, The Compilation Environment) to enable the visibility of these symbols in this header.

The <time.h> header shall declare the structure tm, which shall include at least the following members:

int tm_sec Seconds [0,60].

int tm_min Minutes [0,59].

int tm_hour Hour [0,23].

int tm_mday Day of month [1,31].

int tm_mon Month of year [0,11].

int tm_year Years since 1900.

int tm_wday Day of week [0,6] (Sunday =0).

int tm_yday Day of year [0,365].

int tm_isdst Daylight Savings flag.

The value of tm_isdst shall be positive if Daylight Savings Time is in effect, 0 if Daylight Savings Time is not in effect, and negative if the information is not available.

The <time.h> header shall define the following symbolic names:

NULL

Null pointer constant.

CLOCKS_PER_SEC

A number used to convert the value returned by the clock() function into seconds.

CLOCK_PROCESS_CPUTIME_ID

The identifier of the CPU-time clock associated with the process making a clock() or timer*() function call.

CLOCK_THREAD_CPUTIME_ID

The identifier of the CPU-time clock associated with the thread making a clock() or timer*() function call.

The <time.h> header shall declare the structure timespec, which has at least the following members:

time_t tv_sec Seconds.

long tv_nsec Nanoseconds.

[4] [6] Copyright © 2001-2004 The IEEE and The Open Group, All Rights reserved

The <time.h> header shall also declare the itimerspec structure, which has at least the following members:

struct timespec it_interval Timer period.

struct timespec it_value Timer expiration.

The following manifest constants shall be defined:

CLOCK_REALTIME

The identifier of the system-wide realtime clock.

TIMER_ABSTIME

Flag indicating time is absolute. For functions taking timer objects, this refers to the clock associated with the timer.

CLOCK_MONOTONIC

The identifier for the system-wide monotonic clock, which is defined as a clock whose value cannot be set via clock_settime() and which cannot have backward clock jumps. The maximum possible clock jump shall be implementation-defined.

The clock_t, size_t, time_t, clockid_t, and timer_t types shall be defined as described in <sys/types.h> .

Although the value of CLOCKS_PER_SEC is required to be 1 million on all XSI-conformant systems, it may be variable on other systems, and it should not be assumed that CLOCKS_PER_SEC is a compile-time constant.

The <time.h> header shall provide a declaration for getdate_err.

The following shall be declared as functions and may also be defined as macros. Function prototypes shall be provided.

```
char  *asctime(const struct tm *);
char  *asctime_r(const struct tm *restrict, char *restrict);
clock_t clock(void);
int  clock_getcpuclockid(pid_t, clockid_t *);
int  clock_getres(clockid_t, struct timespec *);
int  clock_gettime(clockid_t, struct timespec *);
int  clock_nanosleep(clockid_t, int, const struct timespec *,
    struct timespec *);
int  clock_settime(clockid_t, const struct timespec *);
char  *ctime(const time_t *);
char  *ctime_r(const time_t *, char *);
double  difftime(time_t, time_t);
struct tm *getdate(const char *);
struct tm *gmtime(const time_t *);
struct tm *gmtime_r(const time_t *restrict, struct tm *restrict);
struct tm *localtime(const time_t *);
struct tm *localtime_r(const time_t *restrict, struct tm *restrict);
time_t  mktime(struct tm *);
int  nanosleep(const struct timespec *, struct timespec *);
size_t  strftime(char *restrict, size_t, const char *restrict,
   const struct tm *restrict);
char  *strptime(const char *restrict, const char *restrict,
        struct tm *restrict);
    time_t  time(time_t *);
    int  timer_create(clockid_t, struct sigevent *restrict,
        timer_t *restrict);
    int  timer_delete(timer_t);
    int  timer_gettime(timer_t, struct itimerspec *);
    int  timer_getoverrun(timer_t);
    int  timer_settime(timer_t, int, const struct itimerspec *restrict,
        struct itimerspec *restrict);
    void  tzset(void);
```

The following shall be declared as variables:

```
    extern int daylight;
    extern long timezone;
    extern char *tzname[];
```

Inclusion of the <time.h> header may make visible all symbols from the <signal.h> header.

Mapping of Time Access Service onto POSIX

TAS Primitive	**POSIX API**	**Comments**
Wall Clock Capability		
TAS_TIME.request and TAS_TIME.indication	clock_gettime()	Recommended. It is also recommended to use the realtime clock, i.e. CLOCK_REALTIME
	time()	Not recommended
Alarm Clock Capability - Blocking		
TAS_ALARM.request and TAS_TIME.indication	clock_nanosleep()	Recommended. It is also recommended to use the realtime clock, i.e. CLOCK_REALTIME
	nanosleep()	Recommended
	sleep()	Not recommended
Alarm Clock Capability - Callback		
TAS_ALARM.request	timer_create(), timer_settime()	Create timer to signal when time expires
TAS_CANCEL_ALARM.request	timer_delete()	
TAS_TIME.indication	Callback function defined using sigaction()	siaction() sets the action performed upon a signal occurring, i.e. define callback function
Metronome Capability		
TAS_METRONOME.request	Not supported	
TAS_CANCEL_METRONOME.request	Not supported	

ANNEX C

INFORMATIVE REFERENCES

[C1] *Procedures Manual for the Consultative Committee for Space Data Systems*. CCSDS A00.0-Y-9. Yellow Book. Issue 9. Washington, D.C.: CCSDS, November 2003.

[C2] *Information technology—Portable Operating System Interface (POSIX)—Part 1: Base Definitions*. International Standard, ISO/IEC 9945-1:2003. 4th ed. Geneva: ISO, 2003.

[C3] *Time Code Formats*. Recommendation for Space Data System Standards, CCSDS 301.0-B-3. Blue Book. Issue 3. Washington, D.C.: CCSDS, January 2002.

[C4] *The Open Group Base Specifications*. Issue 6. IEEE Std 1003.1, 2004 Edition. San Francisco and Piscataway, NJ: The Open Group and IEEE, 2004.

[C5] *Spacecraft Onboard Interface Services*. Report Concerning Space Data System Standards, CCSDS 850.0-G-1. Green Book. Issue 1. Washington, D.C.: CCSDS, June 2007.

[C6] *Spacecraft Onboard Interface Services—Subnetwork Time Distribution Service*. Draft Recommendation for Space Data System Standards, CCSDS 853.0-R-1. Red Book. Issue 1. Washington, D.C.: CCSDS, June 2007.

NOTE – Normative references are listed in 1.4.